IN THE UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF TEXAS
TYLER DIVISION

SYBASE, INC.,	
Plaintiff,	
v.	**Civil Action No. 6:08-cv-24 [LED]**
VERTICA SYSTEMS, INC.,	
Defendant.	

DEFENDANT VERTICA SYSTEMS, INC.'S MOTION FOR SUMMARY JUDGMENT ON INDEFINITENESS

TABLE OF CONTENTS

TABLE OF AUTHORITIES

CASES

STATUTES

OTHER AUTHORITIES

I. ISSUES TO BE DECIDED BY THE COURT

Sybase's patent, U.S. Patent No. 5,794,229[1], has two defects that together render all claims invalid. The first defect is that apparatus claim 16 (and its dependent claims) describe the claimed features only in functional terms, but the specification does not disclose the required "structure" (the specific elements that perform these functions), in violation of the Patent Act. 35 U.S.C. §112, ¶6. The second defect is that method claim 1 (and its dependent claims) suffer from a technical flaw which cannot be "corrected" through claim construction—they require an impossible method step

The '229 patent functionally claims five elements without describing **<u>any</u>** structure that performs those functions. The Patent Act does not allow an inventor to claim all possible structures that perform a function. Rather, an inventor may only claim the specific structures for performing a function that are described in the patent and equivalent structures. If the patent specification only describes functions with "black box" structures, those functions cannot be claimed using the "means plus function" formulation of 35 U.S.C. § 112, ¶6.

Claim 16 of the patent has three "means plus function" claim elements that are described in the specification without structure—only function. They are the "means for creating . . .," "means for transferring . . . ," and "means for forming a vertical page chain . . . ." Dependent claims have additional flawed "means plus function" terms: "data compression means" and "large block transferring means."

In its claim construction brief, Sybase points to many areas in the specification. However, each citation either (a) identifies functional language in the specification, or (b)

[1] U.S. Patent No. 5,794,229 is attached hereto as Exhibit A and is referred to hereinafter as the "'229 patent."

identifies structure that performs other functions, but is not clearly linked to the function actually claimed.

This case involves a software patent and the parties agree that all claimed functions are implemented by one or more software “objects” running on a computer. Portions of some software “objects” are explained in significant detail. For example much of the patent focuses on an “improved” “Buffer Manager” object. The “Buffer Manager” is specifically claimed in related patents (for example, in U.S. Patent 5,794,228), but is not the focus of these claims.

On the other hand, the five “means” at issue in this brief are black-box software “objects” without structure. For the “means for creating” limitation, the existence of a software “object” is mentioned, but even its name is omitted from the patent. For the “means for transferring” limitation, the specification provides the names used to “call” the “transferring” components, but the structure of the “transferring” components is omitted from the ‘229 patent. Similar defects permeate the “means for forming a vertical page chain,” “data compression means” and “large block transferring means.” Sybase incorrectly frames the issue as a debate about whether the structure allegedly disclosed is “sufficient.” However, the question is much simpler—whether or not there is any structure disclosed at all.

At the time Sybase filed its patent application, it had a working commercial product. Thus, the defects in Sybase’s patent are of its own making—Sybase omitted critical structure from its patent application to keep the operation of its commercial embodiment a trade secret. Having made this choice, Sybase cannot now patent those hidden structures using “means plus function” claim language.

The second defect is that the method claims are indefinite due to a drafting error in claim 1. Claim 1 is a method claim containing a number of steps. One step involves storing data from a column of a database table in “at least one” data page. The next step recites “linking together” the data pages. The “linking together” step can only occur when there are at least two data pages. It is impossible to “link together” a single data page. The problem—the fatal flaw—is that the claim does not state that the “linking together” step is practiced only when there are two or more data pages. As worded, the claim requires the “linking together” step to be practiced at all times—including in circumstances when there is only a single data page.

To make sense, claim 1 would need to be rewritten to either (a) change “at least one” to “at least two” or (b) require the “linking” step only if there are “at least two data pages.” But this defect cannot be corrected by this Court through claim construction, rather, this is a matter for the Patent Office, to the extent it is possible at all. The Court’s sole remedy is to declare the claims invalid for indefiniteness.

Because both independent claims suffer from defects rendering them indefinite, the entire patent must be held invalid as indefinite because each dependent claim inherits the flawed attributes from their respective independent claims. Accordingly, Vertica respectfully requests that this Court hold each claim of the ‘229 patent invalid as being indefinite for failing to “conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.” 35 U.S.C. § 112, ¶ 2.

II. THE UNDISPUTED FACTS

Vertica requests that the Court review the Technical Tutorial submitted by Vertica on September 24th for a brief background on databases, database tables, and data pages stored on the database’s storage device.

The following undisputed facts form the basis for this motion:

A. The Defect Of Claim 1

1. Claim 1 is a method claim that includes the steps of storing data from a column of a database table in data pages and then "linking together" all of the data pages. This claim makes sense where there are two or more data pages being "linked together." However, the literal wording of the claim requires the "linking together" of a single data page.

2. Rather than requesting a change to the text of claim 1, the Examiner "interpreted" the claim to mean something different than the literal language selected by the applicant. The Examiner said that he would assume that the database stores data in a single page, or if more than one page is necessary, then in **two or more** pages **that are linked together**: "The linking as claimed, for the purpose of examination is interpreted as follows: It is preferable to store all data that belongs to a particular column of a database in a single memory page, however, for large databases, the data that belongs to a particular column **may not be stored in a single page and it is necessary to store the overflow in a linked page**." (Ex. B, Office Action dated July 18, 1997, p. 6, ¶ 15.)

3. However, the literal language of the claim is defective because it does not limit the "linking together" step to two or more data pages. The claim literally requires the impossible—the "linking together" of a single data page.

B. "Means Plus Function" Limitations Of Independent Claim 16 and Dependent Claims 21 and 23 Lack Corresponding Structure

4. Claim 16 is an apparatus claim that has limitations written in "means plus function" form and which are subject to Section 112, paragraph 6 of the Patent Act.

5. Claim 16 has three "means plus function" limitations at issue here. For each limitation, the function is the text that follows the term "means for:"

(a) "means for creating a vertical partition for each and every column of the database table, each vertical partition having data values for only a single column of the database table." The function of this means plus function limitation is the text recited after the term "means for." As a shorthand, we will refer to this as the "means for creating" limitation.

(b) "means for transferring each vertical partition to and from the storage device, so that at least some of the data values for a particular column are stored together at a contiguous location on the storage device." Again, the function of this limitation is the text recited after the term "means for." As a shorthand, we will refer to this as the "means for transferring" limitation.

(c) "means for . . . forming a vertical page chain for each column, said vertical page chain storing only those data value of the records which correspond to the column for the page chain." The function of this limitation is the text recited after the term "includes." As a shorthand, we will refer to this as the "means for forming a vertical page chain" limitation.

6. Claim 21 includes a "data compression means for compressing each data page according to a type of compression specified on a page-by-page basis." The function of this limitation is the text recited after the term "means for." As a shorthand, we will refer to this as the "data compression means."

7. Claim 23 includes a "large block transferring means for retrieving data values of particular vertical partition as a single large block." The function of this limitation is the text recited after the term "means for." As a shorthand, we will refer to this as the "large block transferring means" limitation.

C. The Figures Do Not Disclose Relevant Structure

8. The patent discloses no special purpose hardware for performing any claimed function. Figure 1A discloses general purpose hardware, but does not disclose any hardware that can perform the specified functions without specialized software.

9. Figure 1B discloses "a block diagram of a software subsystem for controlling the operation of the computer system of FIG. 1A." (Ex. A, '229 patent, col. 4, ll. 64-65; fig. 1B). No algorithms are disclosed in Figure 1B for performing the functions of the means clauses.

10. Figure 2 is a "block diagram of a client/server system in which the present invention is preferably embodied. (*Id.* at col. 4, ll. 66-67; fig. 2). The claims do not require a "client/server system" and no algorithms are disclosed in Figure 2 for performing the functions of the means clauses.

11. Figure 3A discloses prior art and does not disclose the function of the means clauses. (*Id.* at col. 5, ll. 1-3; fig. 3A).

12. Figures 3B and 3C are illustrations of what the invention **does**, but do not show what the invention **is**—no software algorithms are disclosed. Rather, the two figures illustrate that data from each column of a database table is stored in individual pages connected by page pointers to form a page chain. (*Id.* at col. 5, ll. 1-3; col. 13, ll. 12-39; figs. 3B, 3C).

13. Figures 4A-4D and Figure 5 illustrate the results of the functions performed by the compression feature of the invention. These figures do not disclose any software algorithms that would explain how the feature works. (*Id.* at col. 5, ll. 4-13; figs. 4A - 4D).

14. In sum, none of the figures in the patent disclose any structure whatsoever. They illustrate what the invention does, but not how it does it.

D. Software "Objects" In The '229 Patent And Their Structure

15. The system described in the '229 patent employs what is known as "Object Oriented Programming." Software written using Object Oriented Programming differs from conventional software in that, for Object Oriented Programming, the software programmer creates a number of self-contained "objects" that are allowed to interact with each other in some predetermined way. Prior conventional software generally only allowed the programmer to define the series of steps that the software should take. Object-Oriented Programming improved software by imposing more structure upon it. In Object-Oriented Programming, each software

"object" is considered a specific part of a larger software machine, and each has its own structure—much like physical parts of physical machines.

16. Each "object" has, in general, two parts: "data structures" and "methods." A data structure, as the name implies, defines the way the data is stored. A data structure might range from something simple, like a single number or single character, to something complex, like the structure that might hold a complete record of a customer database, including numbers, dates, and text. The "methods" in a software "object" implement the functionality of the object. They might manipulate or modify the data within the object, or permit other objects to access the object's data. For example, a "sort" method might sort data within the object using a specified algorithm. A "find" method might use a particular search algorithm to search through the object's internal data to find a piece of information and send that information to another object. The structure of a "method" is the algorithm that performs the steps necessary to accomplish the method's function.

17. The "structure" of a software "object" therefore is the combination of the data structures of the object and the algorithms implemented by the methods that act on those data structures. For example, the patent discloses a "Buffer Manager" object. The object includes a "Create" method that creates a "Buffer" object. The "Buffer" object, in turn, has a buffer (a data structure) and its own "methods" for transferring data to and from the buffer and disk storage. The patent makes this point at column 17, lines 42-47:

> When starting with a completely empty database, an object of type s_bufman [the Buffer Manager object] is instantiated by the system by invoking the "Create" method of the s_bufman object, s_bufman::Create. This call, in turn, returns a pointer to an object of class s_buf—a buffer or cache. An s_bufman object contains or controls a cache of s_buf.

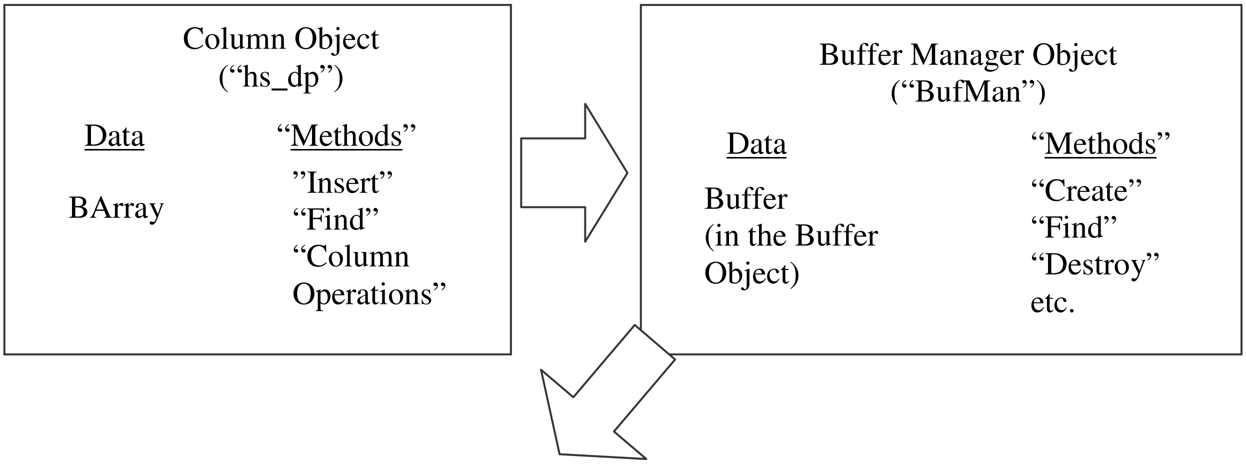
Here, the "s_buf" object is created by the "Create" method in the "s_bufman" object. The created "s_buf" object has a buffer and "Read" and "Write" methods.

18. The algorithm for the "Create" method is disclosed in columns 26 through 27 of the '229 patent. As the Court can see, the algorithm is quite complicated and involved.

19. The "Buffer Manager" also has a "Find" method for searching data in the Buffer object. The algorithm for the "Find" method is shown in columns 30 and 31. The actual algorithms to implement the find functionality is quite involved and spans two columns.

20. There is also a "Column Object" called "hs_dp" (the patent does not say what "hs_dp" stands for). The "Column Object" has an array to hold column data and has methods to insert data into the array, to search for data in the array, and to perform column operations on the data in the array, such as determining the minimum value, maximum value, sum, and average value of the data in the array. (Ex. A, '229 patent, col. 43, l. 36 to col. 45, l. 43.)

21. The "Buffer Manager," "Buffer," and "Column" objects can be represented as follows:



Column Object
("hs_dp")

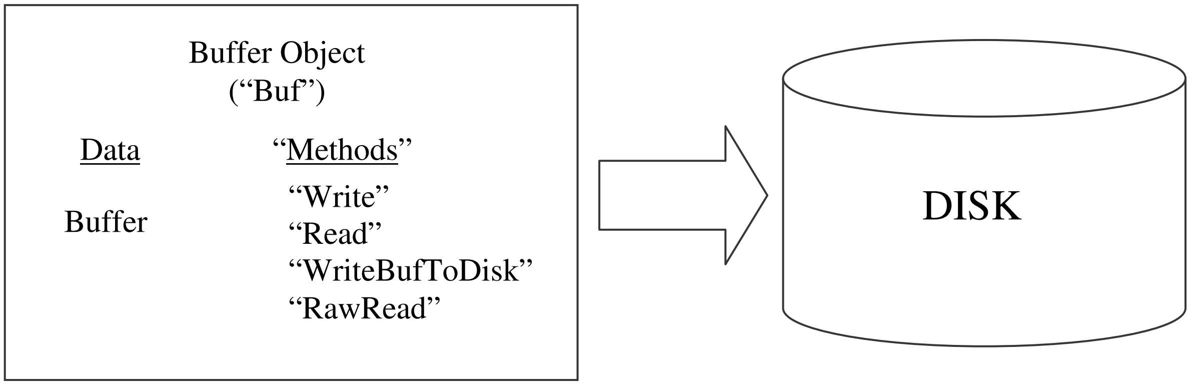
Data
BArray

"Methods"
"Insert"
"Find"
"Column
Operations"

Buffer Manager Object
("BufMan")

Data
Buffer
(in the Buffer
Object)

"Methods"
"Create"
"Find"
"Destroy"
etc.



E. The "Means For Creating A Vertical Partition For Each And Every Column Of The Database Table, Each Vertical Partition Having Data Values For Only A Single Column Of The Database Table" Limitation

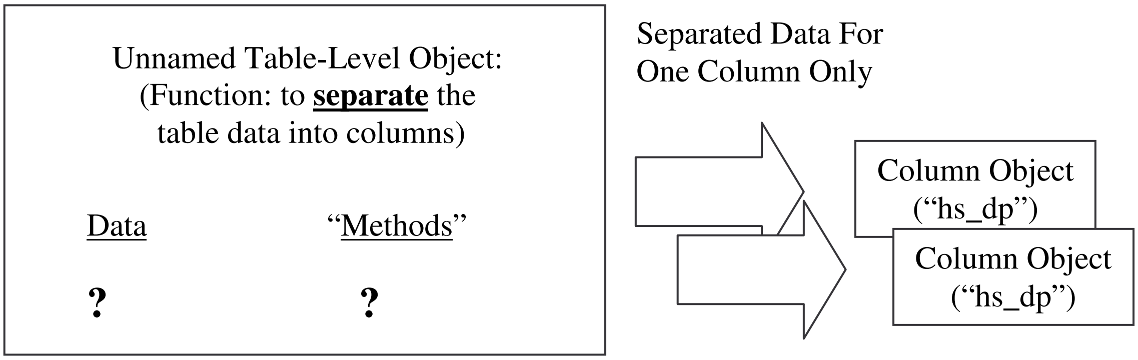
22. The phrase "creating a vertical partition for each and every column of the database table" has an agreed construction which is "separating all data in each column in the database table from any data from any other column in the database table, such that all data in each column can be read from storage without reading any data from any other column." Accordingly, the software that implements the "means for creating" must take row-based table data and **separate** all the columns of that data.

23. There is only one piece of software that is **clearly linked** to the "separating" function. However, the '229 patent merely **mentions** the existence of such an object—but provides no structure for the object. The patent states: "To insert a record into a table, therefore, a **table-level object breaks the data down by column and then hands off each column value** (user data) **to a particular column object** (i.e. object instance of hs_dp class)." (Ex. A, '229 patent, col. 44, ll. 36-40).[2]

24. The structure of this "table-level object" is not disclosed anywhere in the specification. Its data structures were not included in the specification. Its methods were not

[2] Emphasis added throughout this Motion unless otherwise stated.

included in the specification. The patentee hid these details from the public. In fact, the name of the “table-level object” is not even provided. It is an unnamed “black box:”


Unnamed Table-Level Object:
(Function: to separate the table data into columns)
Data
“Methods”
?
?
Separated Data For One Column Only
Column Object (“hs_dp”)
Column Object (“hs_dp”)

25. Sybase cites to the “column objects” in its claim construction brief. However, the “column-level objects” do not perform the separating function. Rather, they receive data that **has already been separated by the unnamed “table-level object.”** The column objects perform other claim functions, such as responding to queries by searching column data—which is the subject of other claimed means plus function limitations.

26. The “column-level objects” and the “buffer manager” are described. But the “table-level object” central to claim 16 is not.

27. Vertica believes it has located this unidentified “Table-Level” software that performs the vertical partitioning in the commercial embodiment of the ‘229 Patent In Suit, and it spans many pages of source code. However, Sybase considers the actual “Table-Level Object” to be highly-confidential source code and it cannot be disclosed here.

F. The "Means For Transferring Each Vertical Partition To And From The Storage Device, So That At Least Some Data Values For A Particular Column Are Stored Together At A Contiguous Location On The Storage Device" Limitation

28. The claim includes a processor, a memory, and a storage device. The "means for transferring" is software running on the processor, and its role is to transfer data between memory and the storage device.

29. The function of the "means for transferring" limitation is to transfer "each vertical partition to and from the storage device, so that at least some data values for a particular column are stored together at a contiguous location on the storage device."

30. The two pieces of software that perform the recited function are the "Write" methods and the "Read" methods inside the Buffer object. The Buffer object ("s_buf") includes the "methods" used to transfer that data between the buffer in memory and disk storage. The software that transfers data **to** disk is the "Write" methods:

> Actual writing of a buffer is done at the level of s_buf, with each buffer (i.e., s_buf object) including an internal Write method for writing itself out, the class method s_buf::Write. . . . . Like the Read method. the Write method invokes subroutine calls to the low-level compression routines or methods.

(Ex. A, '229 patent, col. 20, ll. 9-15).

> The actual write operations are performed by calling s_buf::write (for each s_buf which is "dirty") which in turn calls compression routines.

(*Id.* at col. 34, ll. 22-24).

> WriteBufToDisk, at line 179, undertakes the actual write (i.e., stream) of compressed data to disk.

(*Id.* at col. 42, ll. 63-65).

The software that reads data from the disk is called the "Read Method."

> Complementing the Write methods are the Read methods. Just as the Write methods compressed information on-the-fly, the Read methods decompress information on–the-fly. . . . The Read method for the class is declared at line

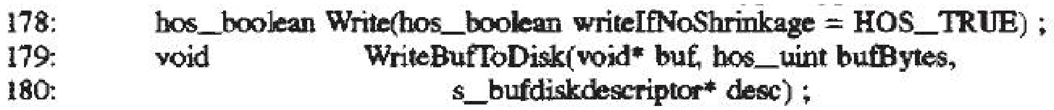
> 148. The method, during operation, invokes the Decompress method, shown at line 70. In conjunction with the Read method, the class also includes a RawRead method at line 146. This method is perhaps best thought of as a counterpart to the WriteBufToDisk method (previously shown at 179). The RawRead method reads the raw data from disk—that is, before decompression is undertaken.

(*Id.* at col. 42, l. 66 to col. 43, l. 12).

31. The actual algorithms that are executed when the Write and Read methods are called are not disclosed in the patent.

32. The function of the "means for transferring" limitation of claim 16 requires that the disclosed "means" ensure that "at least some data values for a particular column **are stored together at a contiguous location on the storage device**." The details of how data is contiguously stored on the storage device are buried in the source code that implements the "Write" and "Read" methods, but are not found in the patent. The patent does not disclose the structure or algorithm that performs this function.

33. The "Write" and "Read" methods are mere "black boxes"—the structure of these methods was not included in the patent application. The patent discloses a single line of source code that identifies each methods' name, for example:



```
178:        hos_boolean Write(hos_boolean writeIfNoShrinkage = HOS_TRUE) ;
179:        void              WriteBufToDisk(void* buf, hos_uint bufBytes,
180:                                    s_bufdiskdescriptor* desc) ;
```

34. The mere disclosure of the method names "Write" and "WriteBufToDisk" does not disclose the structure of the software objects that perform the function claimed in claim 16 of the '229 patent.

G. The Remaining "Means For" Claim Limitations

35. Claim 16 includes the "means for forming a vertical page chain" limitation. The function of the "means for forming a vertical page chain" in claim 16 is "forming a vertical page chain storing only those data value of the records which correspond to the column for the page

chain." The only alleged structure that can perform this function is the software that creates "pointers;" however, the patent does not describe the software for creating pointers. *Aristocrat Technologies Australia Pty Ltd. v. International Game Technology*, 521 F.3d 1328, 1335 (Fed. Cir. 2008) (noting that examples of how data is arranged after processing is "simply examples of the results of the operation of an unspecified algorithm" and thus function, not structure).

36. Claim 21 includes the "data compression means" limitation. The function is "compressing each data page according to a type of compression specified on a page-by-page basis." The alleged structures are the "compress and decompress methods" and commercially available compression/decompression algorithms; however, there is no description of the "compress and decompress methods" or how they interact with the commercial algorithms.

37. Claim 23 includes the limitation, "Large block transferring means" limitation. The function is "retrieving data values of a particular vertical partition as a single large block." Sybase's proposed construction does not list any structure and the patent does not contain any description of the software that performs this function.

III. ARGUMENT

A. Summary Judgment Standard

Summary judgment shall be granted where the moving party "show[s] that there is no genuine issue as to any material fact and that the moving party is entitled to judgment as a matter of law." Fed. R. Civ. P. 56(c); *Celotex Corp. v. Catrett*, 477 U.S. 317, 322 (1986); *Ethicon Endo-Surgery, Inc. v. United States Surgical Corp.*, 149 F.3d 1309, 1315 (Fed. Cir. 1998).

"Indefiniteness under 35 U.S.C. § 112 ¶ 2 is an issue of claim construction and a question of law." *Cordis Corp. v. Boston Scientific Corp.*, 561 F.3d 1319, 1331 (Fed. Cir. 2009). It is thus appropriately resolved on motion for summary judgment.

B. A Means Plus Function Limitation Is Indefinite If The Structure Performing The Function Is Not Adequately Described In The Specification

The Patent Act allows an inventor to claim an invention in terms of what it does rather that what it is. For example, an inventor may claim a "means for fastening" rather than reciting a specific structure such as a bolt, a nail, or a screw. However, an inventor is not allowed to use a "means plus function" limitation to cover **all structures** that can perform the recited function. Instead, the *quid pro quo* for using the "means plus function" option is that the "means plus function" limitation **only covers** the structures actually disclosed in the specification and equivalent structures.

Specifically, 35 U.S.C. § 112, ¶ 6 states: "An element in a claim for a combination may be expressed as a means or step for performing a specified function without the recital of structure, material, or acts in support thereof, and such claim **shall be construed to cover the corresponding structure**, material, or acts **described in the specification** and equivalents thereof." "[I]n return for generic claiming ability [of means plus function terms], the applicant must indicate in the specification what structure constitutes the means." *Biomedino, LLC v. Waters Techs. Corp.*, 490 F.3d 946, 948 (Fed. Cir. 2007). This requirement exists to ensure that patentees are not granted patent rights on purely functional claims. *Aristocrat Technologies Australia Pty Ltd. v. International Game Technology*, 521 F.3d 1328, 1333 (Fed. Cir. 2008).

If the structure is not described in the specification then the "means plus function" limitation cannot be construed and is invalid. Under 35 U.S.C. § 112, ¶ 2, a means plus function limitation without corresponding structure fails to particularly point out and distinctly claim what the applicant regards as his invention.

> [I]f one employs means-plus-function language in a claim, one must set forth in the specification an adequate disclosure showing what is meant by that language. If an applicant fails to set forth an adequate disclosure, the applicant has in effect

> failed to particularly point out and distinctly claim the invention as required by the second paragraph of section 112.

In re Donaldson, 16 F.3d 1189, 1195 (Fed. Cir. 1994) (en banc). If the Court cannot identify the structure that performs the recited function, the claim cannot be construed, and it should be declared indefinite. *See Default Proof Credit Card Sys., Inc. v. Home Depot U.S.A., Inc.*, 412 F.3d 1291, 1302-03 (Fed. Cir. 2005).

It is permissible to use "means plus function" limitations for software inventions. However, the Federal Circuit has repeatedly held that in such cases, the specification must disclose the software algorithms that perform the recited function. *Harris Corp. v. Ericsson, Inc.*, 417 F.3d 1241, 1253 (Fed. Cir. 2005). "[I]n a means-plus-function claim 'in which the **disclosed structure** is a computer, or microprocessor, programmed to carry out an algorithm, the disclosed structure is not the general purpose computer, but rather the special purpose computer programmed to perform the **disclosed algorithm**.'" *Aristocrat*, 521 F.3d at 1333 (quoting *WMS Gaming, Inc. v. Int'l Game Tech.*, 184 F.3d 1339, 1349 (Fed. Cir. 1999)) (emphasis added).

C. The Structure In The Specification Must Be "Clearly Linked" To The Claimed Function In A Means Plus Function Limitation

Sybase points to large swaths of text for the alleged "structure" for the means clauses. Much of the text discloses only functions, and no structure. The remaining text may disclose structure, but discloses structure performing other functions—not the functions at issue here.

The structure that the Court must identify (if it can) is the structure that is "clearly linked" to the functions recited in the claims. "To meet the definiteness requirement, structure disclosed in the specification **must be clearly linked** to and capable of performing the function claimed by the means-plus-function limitation." *Default Proof*, 412 F.3d at 1299 (emphasis added).

Structures that are not clearly linked cannot fulfill the statutory mandate. *B. Braun Med. Inc. v. Abbott Labs.*, 124 F.3d 1419, 1424 (Fed. Cir. 1997) ("structure disclosed in the specification is ‘corresponding’ structure only if the specification or prosecution history clearly links or associates that structure to the function recited in the claim.”). If there is no structure recited in the claim and there is no structure in the specification that is “clearly linked” to the recited function, the claim is indefinite under 35 U.S.C. § 112, ¶ 2. *See*, *e.g.*, *Medical Instrumentation v. Elekta AB.*, 344 F.3d 1205, 1222 (Fed. Cir. 2003).

The requirement for **clear linking** is particularly important in this case because Sybase asserted in the Joint Claim Construction and Prehearing Statement that large portions of the ‘229 patent (spanning many columns) provide the required structures even though—when those columns of text are examined—they have nothing to do with the function **actually claimed** in the ‘229 patent.

Sybase’s position in its claim construction brief is even more perplexing. There, Sybase seeks a “narrative” construction devoid of any citation to the specification at all. Sybase asserts that its refusal to identify the corresponding structure in the specification of the ‘229 patent “will provide more clarity and be more helpful for both the Court and the jury.” (*See* Dkt. No. 107, Sybase’s Claim Construction Brief, p. 32, footnote 13). Sybase’s approach is legally flawed. By statute, the Court must identify the structure in the specification. 35 U.S.C. § 112, ¶ 6 (“such claim shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof”); *In re Donaldson Co.*, 16 F.3d 1189, 1195 (Fed. Cir. 1994) (en banc) (“if one employs means-plus-function language in a claim, one must set forth in the specification an adequate disclosure showing what is meant by that language”).

The obligation of the Court to identify the relevant figures and text (by column and line number) in the specification of the ’229 patent is both important and vital. How else will the fact finder compare the structures in the accused product to the clearly linked structures in the specification? 35 U.S.C. § 112 requires inventors to provide clear guidance on the metes and bounds of the property right staked out by the claims. Without a well-defined boundary around what the patent actually covers, the public notice function of claims would be eviscerated.

The jury must compare the actual structures in Vertica's product to the actual structures disclosed in the specification of the '229 patent. It would be legal error for the jury to compare the structures of Vertica's product to Sybase's vague, functional characterizations of the alleged structures which are divorced from the '229 patent itself. *Ballard Medical Products v. Allegiance Healthcare Corp.*, 268 F.3d 1352, 1361-62 (Fed. Cir. 2001) (noting that a "functional characterization of the scope of the claims is inconsistent with the statutory provision that limits means-plus-function claims to the disclosed structure and equivalents, rather than covering any structure that performs the recited function."). Sybase must cite to actual structure—which it cannot do because the structures at issue in this brief are not disclosed in the patent.

D. Sufficient Structure Must Actually Be Disclosed in the Specification, Not Simply Enabled or Made Obvious

The structure that is identified must be structure actually found in the specification. It is not sufficient to identify a function and state that one of ordinary skill would know how to build the structure. *Aristocrat Technologies Australia Pty Ltd. v. International Game Technology*, 521 F.3d 1328, 1336 (Fed. Cir. 2008) (rejecting the argument that the specification disclosed enough information to allow one of ordinary skill in the art to build the claimed device because it "conflates the requirement of enablement under section 112 paragraph 1 and the requirement to disclose the structure that performs the claimed function under section 112 paragraph 6").

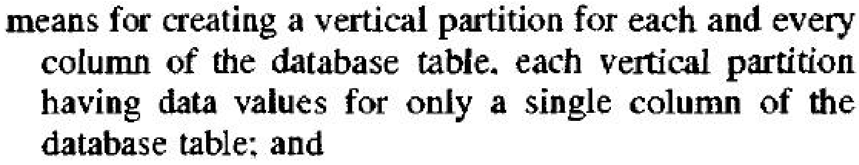
Likewise, it is not sufficient to merely paraphrase the function in the claim and state that one of ordinary skill in the art would know how to build the software. *Blackboard, Inc. v. Desire2Learn, Inc.*, 574 F.3d 1371, 1385 (Fed. Cir. 2009) ("that ordinarily skilled artisans could carry out the recited function in a variety of ways is precisely why claims written in 'means-plus-function' form must disclose the particular structure that is used to perform the recited function.").

E. The "Means For Creating" Limitation Is Indefinite Because The Pertinent "Table-Level Object" Is Not Disclosed

The first step in construing a means-plus-function limitation is to identify the function recited in the claim. *Asyst Techs., Inc. v. Empak, Inc.*, 268 F.3d 1364, 1369 (Fed. Cir. 2001). Once the function is identified (usually a trivial matter), the next step is to identify the structure in the patent specification that performs that function. *Id.*

The recited function for the "means for creating" limitation is stated in the claim: "creating a vertical partition for each and every column of the database table, each vertical partition having data values for only a single column of the database table."

Normally, all parties agree on the recited function—after all, it is explicitly stated in the claim. However, here, there is a dispute that the Court must resolve. Vertica has identified the function recited in the claim itself—which is the text following the "means for":

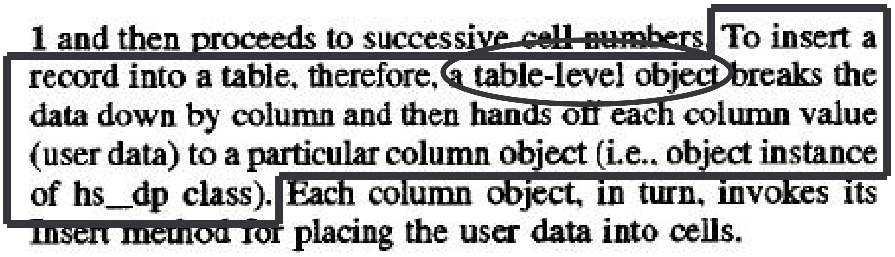


means for creating a vertical partition for each and every column of the database table. each vertical partition having data values for only a single column of the database table; and

Sybase wants to crop the function and only use half of it. There is no basis for doing so. Whatever "means" in the specification is identified, it must perform the entirety of the recited function—it must create a vertical partition for each and every column of the database table and it must ensure that each vertical partition has data values only for a single column of the database table. Sybase cannot read critical functional language out of the claim.

Once the recited function is identified, any pertinent claim constructions must be applied to that function so that its meaning is understood. Here, the parties have agreed that the phrase "creating a vertical partition for each and every column of the database table" means "separating all data in each column in the database table from any data from any other column in the database table, such that all data in each column can be read from storage without reading any data from any other column."

Although Sybase points to various portions of the specification, only **<u>a single sentence</u>** provides any description of software performing this "separating" function—and the sentence merely refers to a mysterious and unnamed "table-level object":



1 and then proceeds to successive cell numbers. To insert a record into a table, therefore, a table-level object breaks the data down by column and then hands off each column value (user data) to a particular column object (i.e., object instance of hs_dp class). Each column object, in turn, invokes its insert method for placing the user data into cells.

Because there is no description of the structure of this nameless "table-level object" for performing the separating function, the "means for creating" limitation is indefinite.

The '229 patent does not identify any structure that is used to separate the tables into distinct columns such that each partition has data from only a single column. All that is disclosed is the existence of a "table-level object" that performs the recited function. The name of the "table-level object" is not provided. The reader cannot tell from the specification what algorithms are implemented by this "table-level object" to separate the columns of a table. The "methods" inside the "table-level object" are not identified. It is a classic—and impermissible—black box.

The case law is clear that if §112, ¶ 6 is not fulfilled, the patent cannot stand. *See*, *e.g.*, *Blackboard, Inc. v. Desire2Learn, Inc.*, 574 F.3d 1371, 1385 (Fed. Cir. 2009). Any other result would clearly and unduly prejudice Vertica and other potential defendants. At trial, the question for the jury will be whether Vertica uses structure that is the same as or equivalent to the structure **disclosed in the specification** of the '229 patent. How is Vertica's table-splitting software to be compared to Sybase's "table-level object" when Sybase failed to disclose the structure of that "table-level object"? It is not possible.[3]

1. The Portions Of The Specification To Which Sybase Cites Do Not Implement The Function Recited In the "Means For Creating" Clause

The sections of the specification that Sybase cites in support of the "means for creating" limitation function relate to other functions – not to the creation of vertical partitions. For example, in its claim construction brief, Sybase asserts that "the patent clearly links code (an hs_dp class) for performing the claimed functions" and cites several sections of the patent. (Dkt. No. 107, Sybase Claim Construction Brief, p. 33). However, the quoted sections of the specification fail to provide the alleged linking. The claimed function is to "**create** a vertical partition for each and every column of the database table" which means to **separate** the table data into columns. The text quoted by Sybase in its brief states that the hs_dp class of objects manipulates vertical partitions that have **already been created**. The partitioning is complete before the "hs_dp" objects see the column data. Nothing in the cited passage suggests that the hs_dp objects "create" vertical partitions.

Rather, the "hs_dp" column objects take the vertical partition already created and perform various tasks on the partitions, such as searching the columns to respond to queries. For example, one "method" in the "hs_dp" objects is the "Find" method. This method is discussed

[3] Vertica notes that while it is not relying on Sybase's commercial embodiment to construe this term, it is significant that the actual "table-level object" appears to span many pages of source code. Vertica is not suggesting that the entirety of the source code should have been copied into the patent, but at the very least a flow chart of the algorithm implemented by the "methods" of the "table-level object" should have been provided.

extensively in the patent. (Ex. A, '229 patent, col. 45, l. 45 to col. 46, l. 4). The "hs_dp" object performs other claimed functions, such as "executing [a] query by scanning only the data pages for [a] particular column." (*Id.* at claim 20). But it does not perform the "creating" function.

Nor can Sybase point to any structure within the hs_dp object that performs the "creating" / "separating" function. On the contrary, Sybase admits on page 33 of its brief that it is the undisclosed "table-level object" that actually performs the function of separating table data into columns. (Dkt. No. 107, Sybase's Claim Construction Brief, pp. 33-34).

Sybase's brief also admits that the '229 patent fails to disclose the relevant algorithm "as a mathematical formula or flow chart" and only "describes the **functions** of the element in prose." (*Id.* at p. 33 (emphasis added)). That is not good enough. *Finisar Corp. v. DirecTV Group, Inc.*, 523 F.3d 1323, 1340 (Fed. Cir. 2008). Sybase, and the Court, must point to specific **structure** in the specification that performs each claimed function. It is **not sufficient** to simply "describe[] the functions" of the software. *Id.* (rejecting alleged support from the specification because it was "nothing more than a restatement of the function"). For Object Oriented software, the structure would include a description of the algorithms implementing the relevant "methods" and a description of the data structures on which the "methods" operate.

Because the structure of the "table-level object" that performs the recited "creating" function is not disclosed in the specification, the "means for creating" is indefinite.

F. The "Means For Transferring" Limitation Is Indefinite Because The Reading And Writing Algorithms Are Not Disclosed

The function for the "means for transferring . . ." limitation is: "transferring each vertical partition to and from the storage device, so that at least some data values for a particular column are stored together at a contiguous location on the storage device."

Again, Sybase desires to prematurely crop the function explicitly recited in the claim such that the disclosed means does not have to ensure that “at least some data values for a particular column are stored together at a contiguous location on the storage device.” However, this portion of the recited function cannot be ignored. To the extent that Sybase identifies structure that does not ensure the recited “**contiguous location on the storage device**” requirement is met, that structure does not meet the explicit claim language.

Likewise, when performing an infringement analysis, any structure identified by Sybase in the accused product has to ensure the contiguity of the data, or this means plus function limitation is not present.

Sybase erroneously asserts that the claim text “contiguous location on the storage device” merely “describes the vertical partition that is transferred” and thus is not part of the function. Sybase is wrong for at least three reasons: First, as a matter of English grammar, the phrase clearly modifies the verb “transferring.” The phrase “so that at least some data values for a particular column are stored together at a contiguous location on the storage device” is not a descriptive clause further modifying the noun “vertical partition,” as Sybase suggests. Rather, the clause is part of the function: “transferring . . . so that at least some data values . . . are stored together.”

Second, there is nothing inherent in the term “vertical partition” that requires that “at least some” data values be stored at a contiguous location. On the contrary, this is a specific functional requirement, found only in claim 16 and some dependent claims.

Third, the final clause is specific to how the transferred data is stored “on the storage device.” It is not describing the vertical partition as it resides in memory before the transfer takes place. Thus, the text of the claim itself precludes the interpretation advocated by Sybase.

The Court should construe the function as the entire clause, "transferring each vertical partition to and from the storage device, so that at least some data values for a particular column are stored together at a contiguous location on the storage device," following the two words "means for."

Now that the function has been properly identified, we must look to the specification to determine what structure, if any, is disclosed that corresponds to the function. The answer is that the structure is not disclosed.

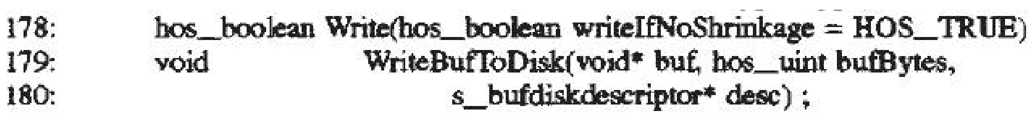
The specification identifies four software "methods" within the Buffer object ("s_buf") that are responsible for transferring information to and from the storage device; however, the specification does not describe an algorithm for transferring data to the storage device so that some data values "are stored together at a contiguous location on the storage device."

Sybase, in its Claim Construction brief, asserts that the Buffer Manager performs the functions of reading and writing data pages to and from a storage device. (Dkt. No. 107, Sybase's Claim Construction Brief, p. 35). However, the actual structure disclosed in the patent is quite different. According to the patent, the Buffer Manager object (the "s_bufman") has a "Create" method that creates a Buffer object ("s_buf"). The Buffer object has its own "methods," and those "methods" actually do the transferring to and from disk. The '229 patent specifically states that the "Create" method of the buffer manager does not do the actual writing. (Ex. A, '229 patent, col. 17, ll. 62-65). Thus, to be precise—and precision is important when identifying the structure in the specification—the Buffer Manager does **<u>not</u>** handle the "transferring" function at all. Rather, the Buffer object handles that function itself.

Sybase then correctly points to the “Read,” “RawRead,” “Write,” and “WriteBufToDisk” functions, **which are part of the s_buf object**, as the structure that performs the claimed function. (Ex. A, ‘229 Patent, col. 42, l. 58 to col. 43, l. 12).

Thus, the question is: Do the “Read”, “RawRead,” “Write,” and “WriteBufToDisk” methods of the s_buf object ensure “that at least some data values for a particular column are stored together at a contiguous location on the storage device,” and if so, where is the disclosure of the algorithms to perform this function? The specification provides no guidance to answer these questions, and thus claim 16 is indefinite.

Of all the text identified by Sybase as being structure for the “means for transferring” clause, only the references to the Read and Write software “methods” relate to the actual function claimed—transferring data to and from the storage device. But the details of those “methods” are **not** disclosed—only the names are provided and the variables passed into and out of the methods, for example:



```
hos_boolean Write(hos_boolean writeIfNoShrinkage = HOS_TRUE) ;
void            WriteBufToDisk(void* buf, hos_uint bufBytes,
                        s_bufdiskdescriptor* desc) ;
```

There is no disclosure here about how the Write software actually works or how it accomplishes its goal. The names (here, “Write()” and “WriteBufToDisk()”) do not answer the question of how the “methods” accomplish their reading and writing functions.[4]

All that is revealed is that the user can call a routine named “Write” and when called that software will execute an undisclosed algorithm that will write the data from a buffer in memory

[4] These software statements are “declarations.” A declaration is a statement indicating that a piece of software (including the algorithms performed by the software) will appear **elsewhere** in the source code. A declaration does not do anything. In fact, much of the “software” disclosed in the ‘229 patent are declarations that do nothing. In a nutshell, much of the “software” disclosed in the patent cannot be “run” and one cannot determine what the software does—that is, what the algorithms actually are—by reading the declarations.

to the disk (the storage device). The text of the specification provides no more guidance as to the algorithms executed when these Write and Read software components are run.

Sybase's mere naming of a software object that allegedly performs a recited function is legally insufficient for a means-plus-function limitation. It does nothing more than define the metes and bounds of "a black box that performs a recited function." *Blackboard, Inc. v. Desire2Learn, Inc.*, 574 F.3d 1371, 1383 (Fed. Cir. 2009). Such "definitions" do not disclose enough information to satisfy 35 U.S.C. § 112, as "[t]here is nothing in the written description that expressly states what is going on inside [the] computer." *Net MoneyIN, Inc. v. VeriSign, Inc.*, 545 F.3d 1359, 1367 n.3 (Fed. Cir. 2008).

Large sections of the specification that Sybase cites have nothing to do with the reading and writing of data to storage. For example, Sybase points to columns of text describing aspects of the Buffer Manager that do not pertain to the Read and Write methods at issue here. Those citations are not relevant because they do not concern the claimed function—which is "transferring each vertical partition to and from the storage device, so that at least some data values for a particular column are stored together at a contiguous location on the storage device." The '229 patent states that the transferring is handled by the "Read" and "Write" methods of the s_buf objects and not by the Buffer Manager, which instead contains functions to properly maintain the buffers themselves. (Ex. A, '229 Patent, col. 18, ll. 42-49; col. 19, ll. 4-25; col. 20, ll. 9-15; col. 34, ll. 22-24).

Because the algorithms underlying the "read" and "write" methods are not disclosed in the '229 patent, the "means for transferring" limitation is indefinite and claim 16 and its dependent claims are thus invalid.

G. Three Additional "Means For" Claim Limitations Are Also Indefinite

There are three additional "means for" claim limitations that are indefinite for the same reasons described above, namely, that the patent does not describe the software that performs these functions. Vertica notes that if the Court grants Vertica's motion of indefiniteness as to the previous two "means plus function" clauses and declares claim 16 and its dependent claims invalid on that basis, the Court need not address these remaining terms.

The remaining term in claim 16 is "means for forming a vertical page chain" and its function is "forming a vertical page chain storing only those data value of the records which correspond to the column for the page chain." Sybase identifies four elements as the alleged structure: pointers, a system catalog, a B-Tree, and a block array. Only pointers can actually form page chains. The other three elements are indexes that can point to a pages in a database, but they cannot be used to form a page chain.[5]

The page chains disclosed in the patent are illustrated in Figures 3A-3C. The page chain is a series of pages that are connected—one after the other—by references in the pages pointing to successive pages to form the "chain." The disclosed pointers are the links that connect individual pages in a page chain. The structure for "forming" a page chain is therefore the software that creates the page pointers to link the various pages together. However, the patent does not describe the software that connects the pages with pointers. There is no algorithm in the patent showing how the pages are chained together to form a page chain.

The function of the "data compression means" in claim 21 is "compressing each data page according to a type of compression specified on a page-by-page basis." Sybase identifies two elements of alleged structure: (a) the "compress and decompress methods" and (b)

[5] The system catalog, B-Tree and block array do not perform the "chaining" function and those features are not "clearly linked" to the chaining function. A complete explanation of why Sybase is incorrect is found in Vertica's Claim Construction Brief, filed with this Motion.

commercially available compression/decompression algorithms. These commercial compression algorithms were admittedly in the prior art. However, Sybase contends that this “means” is something new, and thus no prior art structure could perform this function.

Sybase’s argument appears to be that the “compress and decompress” methods use these commercially available algorithms in some unique and patentable way. However, the claim is indefinite because there is no description of—much less an algorithm for—the “compress and decompress methods” or how they interact with the off-the-shelf compression algorithms.

The function of the “large block transfer means” in claim 21 is “retrieving data values of a particular vertical partition as a single large block.” Sybase’s proposed construction does not identify any structure. Sybase states, “[t]he corresponding structure is a computer with source code for retrieving data values of a particular vertical partition as a single large block.” As a matter of law, such an identification—“an algorithm executed by a computer”—is insufficient. *Harris Corp. v. Ericsson, Inc.*, 417 F.3d 1241, 1253 (Fed. Cir. 2005). In its brief, Sybase points to a line in the specification that identifies the buffer managers as being capable of large block transfers, “[t]he retrieval itself can be done using more-efficient large block I/O transfers.” (Dkt. No. 107, Sybase’s Claim Construction Brief, p. 44). But there is no disclosure of the software (or algorithm) within the buffer managers that performs this function. As discussed above, the explanation for how data is retrieved from the storage device is buried in the source code that implements the “Read” methods—source code that is in the commercial embodiment of the patent, but is not disclosed in the patent itself.

H. The Phrase “Linking Together The At Least One Data Page” In Claim 1 Makes No Sense And Is Indefinite

The Court should find that claim 1 and each of its dependent claims is indefinite because each requires a step that is not possible: “linking together all of said at least one data page.” This is a technical defect that cannot be corrected by the Court. Rather, this defect must be corrected, if correction is even possible, by the United States Patent and Trademark Office.

Claim 1 is a method claim that has a step of "storing the data value for each column in the at least one data page associated with the column." There is no dispute that "at least one" means one or more. Therefore the data might be stored in a single page or data might be stored in multiple pages. Both possibilities are within the scope of this claim.

However, the next step in the method is "linking together all of said at least one data pages." The plain meaning of "linking **together**" means to link at least two things. It makes no sense to "link together" a single thing.

The patent specification only discloses the linking of **two or more** pages, (*See*, *e.g.*, Ex. A, '229 patent, figs. 3B and 3C). There is no disclosure of linking a page to itself, and grammatically speaking, it is not possible to link **together** a single page. The adverb "together" requires at least two things. (*See* Ex. C, "Together," Random House Webster's College Dictionary, (1995)).

The claim is defective. Furthermore, it is unclear how the claim should be corrected. Two possibilities are:

(1) The phrase "at least one" throughout claim should be changed to require "at least two." This would be consistent with the specification, which only discloses linking together two or more data pages.

(2) The claim should be changed to allow one page, but state that the linking together step would only be practiced if two or more data pages are used. Thus, the claim might be rewritten to say something like "only if two or more pages are present, linking together all of

said data pages . . . ." The Examiner was apparently stumped by the literal text of claim 1 and therefore chose to "reinterpret" claim 1 along the lines of this second option:

> The linking as claimed, for the purpose of examination is interpreted as follows: It is preferable to store all data that belongs to a particular column of a database in a single memory page, however, when for large databases, the data that belong to a particular column may not be stored in a single page and it is necessary to store the overflow in a linked page(.)

(Ex. B, Office Action dated July 18, 1997, p. 6, ¶15).

The problem with this interpretation is that it bears no resemblance to the actual claim language. The Examiner's commentary which does not track the actual claim language must be ignored. *Salazar v. Procter & Gamble Co.*, 414 F.3d 1342, 1347 (Fed. Cir. 2005) ("An examiner's statement cannot amend a claim"). This Court is permitted to construe ambiguous claims in light of the intrinsic record, but it is not free to rewrite them. *Quantum Corp. v. Rodime, PLC*, 65 F.3d 1577, 1584 (Fed. Cir. 1995) ("[I]t is well settled that no matter how great the temptations of fairness or policy making, courts do not redraft claims").

Moreover, a court may correct an error in a patent, but only if (1) the correction is not subject to reasonable debate based on consideration of the claim language and the specification and (2) the prosecution history does not suggest a different interpretation of the claims. *Novo Indus. L.P v. Micro Molds Corp.*, 350 F.3d 1348, 1354, 1358 (Fed. Cir. 2003) ("[I]n order to make sense out of the patent, the district court was required to guess as to what was intended. That is beyond its authority.") Here we have substantive changes to the claims and not obvious typographical or clerical mistakes that the Court may correct. *See*, *id.* at 1357 ("The present case does not fall within the ambit of the district court's authority, for the nature of the error is not apparent from the face of the patent.").

Because the claim as written makes no sense due to a technical defect and the "correct" meaning of the claim cannot be determined, the claim is insolulably ambiguous and indefinite. *Honeywell Intern., Inc. v. International Trade Com'n*, 341 F.3d 1332, 1340-41 (Fed. Cir. 2003).

In its claim construction brief, Sybase is unable to explain how the "linking" step is performed with a single data page. Sybase first argues that claim 1 clearly and repeatedly uses the phrase "at least one" data page. This is not in dispute. The problem is that the phrase "linking **together** all of said at least one data page" makes no sense.

Sybase next argues that the Examiner "understood exactly what Sybase was claiming." Again, Vertica concedes that the Examiner provided an interpretation of the "linking" step. The problem is that the Examiner's interpretation is not grounded in the language of the claim itself. The Examiner went far beyond "claim construction" and examined a claim having different words than the actual claim. The actual claim says nothing about linking only if more than one page is used. The claim as written requires "linking together" even when there is a single data page. Accordingly, as written, claim 1 and its dependent claims are defective, and the Court should declare them invalid.

IV. CONCLUSION

Because the '229 patent does not disclose the structure for the "means for creating" or "means for transferring" limitations of claim 16, claims 16 and its dependent claims (17 through 24) are invalid as indefinite. Because claim 1 of the '229 patent has a technical defect—it is impossible to "link together" a single data page—claim 1 and its dependent claims (2 through 15) are invalid as indefinite. For these reasons, and the other reasons expressed herein, Vertica respectfully requests that this Court GRANT its Motion on Indefiniteness.

Dated: October 13, 2009

Respectfully submitted,

FISH & RICHARDSON P.C.

By: */s Steven R. Katz*

Steven R. Katz, BBO#642732
FISH & RICHARDSON P.C.
225 Franklin Street
Boston, MA 02110-2804
617-542-5070 (Telephone)
617-542-8906 (Facsimile)

Ralph Ritch Roberts III
State Bar No. TX 24041794
Roberts@fr.com
FISH & RICHARDSON P.C.
1717 Main St., Suite 5000
Dallas, TX 75201
(214) 747-5070 (Telephone)
(214) 747-2091 (Facsimile)

Counsel for
VERTICA SYSTEMS, INC.

CERTIFICATE OF SERVICE

The undersigned hereby certifies that a true and correct copy of the above and foregoing document has been served on all counsel of record who are deemed to have consented to electronic service via the Court's CM/ECF system per Local Rule CV-5(a)(3). Pursuant to the Court's Order, courtesy copies have been delivered to the Court via hand delivery and shipped to the Technical Advisor for overnight delivery.

<u>/s Steven R. Katz</u>
Steven R. Katz

22284219.doc